Protocolli a chiave pubblica

Crittografia

Luciano Margara

Unibo

2022

Cenni Storici

Nel 1976 Diffie e Hellman, e indipendentemente Merkle, introdussero un nuovo concetto che avrebbe rivoluzionato il modo di concepire le comunicazioni segrete: i cifrari a chiave pubblica.

Chiavi

Nei cifrari simmetrici visti sinora la chiave di cifratura è uguale a quella di decifrazione (o comunque ciascuna può essere facilmente calcolata dall'altra), ed è nota solo ai due partner che la scelgono di comune accordo e la mantengono segreta.
Nei cifrari a chiave pubblica, o asimmetrici, le chiavi di cifratura e di decifrazione sono completamente diverse tra loro. Esse sono scelte dal destinatario che rende pubblica la chiave di cifratura $k[pub]$, che è quindi nota a tutti, e mantiene segreta la chiave di decifrazione $k[prv]$ che è quindi nota soltanto a lui.

Chiavi

Esiste una coppia $k[pub]$, $k[prv]$ per ogni utente del sistema, scelta da questi nella sua veste di possibile destinatario $Dest$. La cifratura di un messaggio m da inviare a $Dest$ è eseguita da qualunque mittente come $c = C(m, k[pub])$, ove sia la chiave $k[pub]$ che la funzione di cifratura C sono note a tutti. La decifrazione è eseguita da $Dest$ come $m = D(c, k[prv])$, ove D è la funzione di decifrazione anch'essa nota a tutti, ma $k[prv]$ non è disponibile agli altri che non possono quindi ricostruire m.

Asimmetria

L'appellativo di asimmetrici assegnato a questi cifrari sottolinea i ruoli completamente diversi svolti da *Mitt* e *Dest*, in contrapposizione ai ruoli intercambiabili che essi hanno nei cifrari simmetrici ove condividono la stessa informazione (cioè la chiave) segreta.

Proprietà

Per ogni possibile messaggio m si ha:

$$D(C(m, k[pub]), k[prv]) = m$$

Ossia *Dest* deve avere la possibilità di interpretare qualunque messaggio che gli altri utenti decidano di spedirgli.

Proprietà

La sicurezza e l'efficienza del sistema dipendono dalle funzioni C e D, e dalla relazione che esiste tra le chiavi $k[prv]$ e $k[pub]$ di ogni coppia.

La coppia $k[prv]$ e $k[pub]$ è facile da generare, e deve risultare praticamente impossibile che due utenti scelgano la stessa chiave.

Proprietà

Dati m e $k[pub]$, è facile per il mittente calcolare il crittogramma $c = C(m, k[pub])$.

Dati c e $k[prv]$, è facile per il destinatario calcolare il messaggio originale $m = D(c, k[prv])$

Pur conoscendo il crittogramma c, la chiave pubblica $k[pub]$, e le funzioni C e D, è difficile per un crittoanalista risalire al messaggio m.

Facile e Difficile

I termini "facile" e "difficile" devono intendersi in senso computazionale.

Funzioni one-way

C deve essere una funzione one-way, cioè facile da calcolare e difficile da invertire, ma deve contenere un meccanismo segreto detto trap-door che ne consenta la facile invertibilità solo a chi conosca tale meccanismo. La conoscenza di $k[pub]$ non fornisce alcuna indicazione sul meccanismo segreto, che è svelato da $k[prv]$ quando questa chiave è inserita nella funzione D.

Richiami di algebra modulare

Preso un numero intero positivo n indichiamo con $\mathbb{Z}_n = \{0, 1, \ldots, n-1\}$ l'insieme di tutti gli interi non negativi minori di n

Esempio: $n = 6$ e $\mathbb{Z}_6 = \{0, 1, 2, 3, 4, 5\}$

Richiami di algebra modulare

Quindi, Zn* è l'insieme degli interi minori di n che non condividono nessun divisore comune con n (ad eccezione del numero 1).

Preso un numero intero positivo n indichiamo con $\mathbb{Z}_n^*$ l'insieme degli elementi di $\mathbb{Z}_n$ relativamente primi con n (0 escluso, 1 incluso)

a è relativamente primo con b se e solo se $MCD(a, b) = 1$

Richiami di algebra modulare

Esempi

$\mathbb{Z}_6^* = \{1, 5\}$ non c'é il 2, 3 e il 4 perché composti da numeri che dividono 6

$\mathbb{Z}_9^* = \{1, 2, 4, 5, 7, 8\}$ non c'é il 3 e il 6 perché composti da numeri che dividono 9

$\mathbb{Z}_{11}^* = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$

Richiami di algebra modulare

Il problema di determinare $\mathbb{Z}_n^*$ dato n è in genere computazionalmente difficile poiché richiede tempo proporzionale al valore di n (per esempio per confrontare con n tutti gli elementi di $\mathbb{Z}_n$), quindi esponenziale nella sua dimensione, cioè nel numero di bit con cui si rappresenta n. Se però n è un numero primo si ha direttamente $\mathbb{Z}_p^* = \{1, \ldots, p-1\}$. quindi non é piú esponenziale

Richiami di algebra modulare

$a \mod n$ è il resto della divisione intera tra a e n

$a \equiv b \mod n$ se e solo se $a \mod n = b \mod n$

$a \equiv b \mod n$ se e solo se $a = b + kn$

Richiami di algebra modulare

$$(a + b) \mod n = ((a \mod n) + (b \mod n)) \mod n$$

$$(a - b) \mod n = ((a \mod n) - (b \mod n)) \mod n$$

$$(a \times b) \mod n = ((a \mod n) \times (b \mod n)) \mod n$$

$a^{r \times s} \mod n = ((a^r \mod n)^s) \mod n$, con r e s interi positivi qualunque

Funzione di Eulero

$$\Phi(n) = |\mathbb{Z}_n^*|$$

Calcola la Cardinalità di Zn*

$\Phi(p) = p - 1$

p e q = numeri primi distinti

$\Phi(pq) = (p-1)(q-1)$

a primo con b allora $\Phi(ab) = \Phi(a)\Phi(b)$

$\Phi(n)$ difficile da calcolare

La funzione di Eulero può essere difficile da calcolare per numeri n grandi, soprattutto quando n non è un numero primo, ma una fattorizzazione di numeri primi.

Funzione di Eulero

Per ogni $p > 1$ primo e per ogni $k \geq 1$ intero

$$\begin{aligned} \Phi(p^k) &= \Phi(p)p^{k-1} \\ &= (p-1)p^{k-1} \end{aligned}$$

Funzione di Eulero

Se è nota la scomposizione in fattori primi di n allora $\Phi(n)$ è facile da calcolare.

$$n = p_1^{m_1} \cdot \dots \cdot p_k^{m_k}$$

$$\begin{aligned} \Phi(n) &= \Phi(p_1^{m_1} \cdot \dots \cdot p_k^{m_k}) \\ &= \prod_{i=1}^{k} \Phi(p_i^{m_i}) \\ &= \prod_{i=1}^{k} (p_i - 1) p_i^{m_i - 1} \end{aligned}$$

Funzione di Eulero: Esempio

Sia $n = 2^3 3^2 11 = 792$

$$\begin{aligned} \Phi(792) &= \Phi(2^3)\Phi(3^2)\Phi(11) \\ &= 4 \cdot 6 \cdot 10 \\ &= 240 \end{aligned}$$

Teorema di Eulero (uno dei tanti)

è un intero positivo

(è un intero che è relativamente primo a n)

Sia $n > 1$ e a ~~primo con n~~ allora:

$$a^{\Phi(n)} \equiv 1 \mod n$$

numero di numeri tra 1 e n−1 che sono relativi primi a n.

congruente

Piccolo teorema di Fermat

Sia a primo con p

relativamente

numero primo

$$a^{p-1} \equiv 1 \mod p$$

congruente

Funzioni one-way con trap-door

Una funzione è one-way con trap-door se è facile da calcolare e difficile da invertire a meno che non si conosca qualche informazione aggiuntiva che ne faciliti l'inversione.

Funzioni one-way con trap-door

Meccanismi di tipo one-way con trap-door si incontrano con frequenza in ogni campo. Il più ovvio può essere messo in relazione alla protezione "domestica" delle comunicazioni. I messaggi sono comuni lettere, la chiave pubblica è l'indirizzo del destinatario, il metodo pubblico di cifratura consiste nell'imbucare una lettera nella cassetta postale del destinatario, accessibile a tutti. Il metodo privato di decifrazione consiste nell'apertura della cassetta mediante la chiave (metallica) posseduta solo dal destinatario. Facile depositare le lettere nella cassetta ma difficile estrarle se non si possiede la chiave.

Funzioni one-way con trap-door

Altro esempio "fisico" di funzione one-way con trapdoor: Lucchetto della bicicletta.

Funzioni one-way con trap-door

Esistono funzioni one-way con trap-door in matematica ?

Funzioni one-way con trap-door

Sono state individuate alcune funzioni che possiedono i requisiti richiesti utilizzando proprietà della teoria dei numeri e dell'algebra modulare. Il calcolo diretto di queste funzioni è infatti semplice e la loro inversione è semplice solo se si dispone di una informazione aggiuntiva sui dati, cioè di una chiave privata. Senza questa informazione l'inversione richiede la soluzione di un problema NP-Hard, o comunque di un problema per cui non si conosce un algoritmo polinomiale: tale inversione rimarrà quindi computazionalmente difficile fino all'improbabile momento in cui si trovi una soluzione efficiente a questi problemi.

Fattorizzazione

numeri primi

Dati p e q calcolare $n = pq$ è facile.
Dato n trovare p e q è difficile, ma se si conosce p oppure q diventa facile.

Fattorizzare n non è stato dimostrato essere un problema NP-Hard, ma è ritenuto da tutti un problema difficile (complessità esponenziale).

`https://web.archive.org/web/20061209135708/http://www.rsasecurity.com/rsalabs/node.asp?id=2093`

Calcolo della radice in modulo

Calcolare la potenza $y = x^z \mod s$, con x, z, s interi, richiede tempo polinomiale se si procede per successive esponenziazioni.

Se s non è primo e se ne ignora la fattorizzazione, invertire la funzione, cioè calcolare $x = \sqrt[z]{y} \mod s$ se sono noti y, z, s richiede tempo esponenziale per quanto noto fino a oggi (lo status computazionale di questo problema è simile a quello della fattorizzazione)

Calcolo della radice in modulo

Se però x è primo con s e si conosce v tale che $zv \equiv 1 \mod \Phi(s)$, si ha:

$$\begin{aligned} y^v \mod s &= x^{zv} \mod s \\ &= x^{1+k\Phi(s)} \mod s \\ &= x \mod s \end{aligned}$$

in cui l'ultima eguaglianza è basata sul teorema di Eulero. Si ricostruisce quindi in tempo polinomiale x calcolando $y^v \mod s$. In questo caso v è la chiave segreta per invertire la funzione.

Calcolo del logaritmo discreto

Calcolare la potenza $y = x^z \mod s$.

Calcolare z

Vantaggi dei protocolli chiave pubblica

Se gli utenti di un sistema sono n, il numero complessivo di chiavi (pubbliche e private) è $2n$ anziché $n(n-1)/2$.

Non è richiesto alcuno scambio segreto di chiavi tra gli utenti

Difetti dei protocolli chiave pubblica

Il sistema è esposto ad attacchi del tipo chosen plain-text in modo del tutto ovvio. Un crittoanalista può scegliere un numero qualsiasi di messaggi in chiaro $m_1, m_2, \ldots, m_h$ e cifrarli utilizzando la funzione pubblica C e la chiave pubblica $k[pub]$ di un destinatario $Dest$, ottenendo così i crittogrammi $c_1, c_2, \ldots, c_h$.

Difetti dei protocolli chiave pubblica

A questo punto, spiando sul canale di comunicazione, egli può confrontare qualsiasi messaggio cifrato c^* in viaggio verso *Dest* con i crittogrammi di cui è in possesso: se c^* coincide con uno di essi il messaggio è automaticamente decifrato; se invece $c^* \neq c_i$, per ogni i, il crittoanalista ha comunque acquisito una informazione importante, cioè che il messaggio è diverso da quelli che lui ha scelto. L'attacco è particolarmente pericoloso se il crittoanalista sospetta che *Dest* debba ricevere un messaggio particolare ed è in attesa di vedere quando questo accada; oppure se c^* rappresenta un messaggio breve e di struttura prevedibile, come per esempio un indirizzo Internet o una password scelta ingenuamente.

Difetti dei protocolli chiave pubblica

Questi sistemi sono molto più lenti di quelli basati su cifrari simmetrici: stime indicano che il rapporto tra le loro velocità sia da due a tre ordini di grandezza. Si potrebbe obiettare che questo è un problema secondario a causa della continua crescita di velocità dei calcolatori, ma l'effetto è invece sensibile per la richiesta sempre crescente di comunicazioni sicure.

Cifrario proposto da Merkle

Il primo cifrario, proposto da Merkle, basava la difficoltà di inversione della funzione C sulla risoluzione del problema dello zaino. Benché tale problema sia NP-Hard il cifrario è stato violato per altra via, mostrando ancora una volta con quanta cautela vada affrontato il problema della sicurezza (successivi cifrari basati sullo stesso problema sono invece rimasti inviolati).

Cifrario RSA

Il secondo cifrario, proposto da Rivest, Shamir e Adleman (1978) e noto come RSA, fonda la sua sicurezza sulla difficoltà di fattorizzare grandi numeri interi. Benché tale problema non sia dimostratamente NP-Hard, e quindi potrebbe essere "più semplice" del problema dello zaino, RSA è sostanzialmente inviolabile per chiavi sufficientemente lunghe, ed è il cifrario asimmetrico di più largo impiego.

Cifrario RSA: creazione delle chiavi

Come possibile destinatario, ogni utente $Dest$ esegue le seguenti operazioni:

1. sceglie due numeri primi p, q molto grandi
2. calcola $n = pq$ e $\Phi(n) = (p-1)(q-1)$
3. sceglie un intero e minore di $\Phi(n)$ e primo con esso
4. calcola l'intero d inverso di e modulo $\Phi(n)$
5. rende pubblica la chiave $k[pub] = (e, n)$, e mantiene segreta la chiave $k[prv] = (d)$.

Cifrario RSA: il messaggio

Ogni messaggio è codificato come sequenza binaria, che viene trattata come un numero intero m. Per impiegare il cifrario deve risultare $m < n$, il che è sempre possibile dividendo il messaggio in blocchi di al più $\lfloor \log_2(n) \rfloor$ bit.

Cifrario RSA: codifica

$$c = m^e \mod n$$

Cifrario RSA: decodifica

$$m = c^d \mod n$$

Cifrario RSA: esempio

- $p = 5$ e $q = 11$
- $n = 55$ e $\Phi(n) = 40$
- $e = 7$ (primo con 40)
- $d = 23$ $(23 \times 7 \equiv 1 \mod 40)$
- $k[pub] = (7, 55)$ e $k[prv] = (23)$
- $c = m^7 \mod 55$
- $m = c^{23} \mod 55$

Cifrario RSA: esempio

m	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
m^7	0	1	18	42	49	25	41	28	2	4	10	11	23	7	9	5	36	8	17	24	15	21	33	12	29	20	16
m^{23}	0	1	8	27	9	15	51	13	17	14	10	11	23	52	49	20	26	18	2	39	25	21	33	12	19	5	31
m^{161}	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26

m	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54
m^7	3	52	39	35	26	43	22	34	40	31	38	47	19	50	46	48	32	44	45	51	53	27	14	30	6	13	37	54
m^{23}	48	7	24	50	36	43	22	34	30	16	53	37	29	35	6	3	32	44	45	41	38	42	4	40	46	28	47	54
m^{161}	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54

Cifrario RSA: Correttezza

Per qualunque intero $m < n$ si ha:

$$(m^e \mod n)^d \mod n = m$$

dove n, e, d sono i parametri del cifrario RSA.

Cifrario RSA: Correttezza

$$(m^e \mod n)^d \mod n = m^{ed} \mod n$$

Distinguiamo 2 casi:
Caso 1. p e q non dividono m
Caso 2. p (oppure q) divide m, ma q (oppure p) non divide m

p e q non dividono m

Abbiamo $\gcd(m, n) = 1$, quindi per il teorema di Eulero risulta $m^{\Phi(n)} \equiv 1 \mod n$. Poiché d è l'inverso di e modulo $\Phi(n)$, abbiamo $ed \equiv 1 \mod \Phi(n)$, ovvero $ed = 1 + r\Phi(n)$, con r intero positivo opportuno. Otteniamo quindi:

$$\begin{aligned} m^{ed} \mod n &= m^{1+r\Phi(n)} \mod n \\ &= m(m^{\Phi(n)})^r \mod n \\ &= m1^r \mod n \\ &= m \end{aligned}$$

p divide m, ma q non divide m

Poiché p divide m abbiamo $m \equiv m^r \equiv 0 \mod p$, ovvero $(m^r - m) \equiv 0 \mod p$, per qualunque intero positivo r. Analogamente al caso precedente abbiamo:

$$\begin{aligned} m^{ed} \mod q &= m^{1+r\Phi(n)} \mod q \\ &= m\, m^{r(p-1)(q-1)} \mod q \\ &= m\left(m^{q-1}\right)^{r(p-1)} \mod q \\ &= m \mod q \end{aligned}$$

l'ultima uguaglianza è dovuta alla relazione $m^{q-1} \equiv 1 \mod q$ per il teorema di Eulero

p divide m, ma q non divide m

Dunque $m^{ed} \equiv m \mod q$, e quindi $(m^{ed} - m) \equiv 0 \mod q$. Ne consegue che $m^{ed} - m$ è divisibile sia per p che per q, quindi è divisibile per il loro prodotto $pq = n$. Ovvero $(m^{ed} - m) \equiv 0 \mod n$ da cui deriva immediatamente la tesi.

p e q dividono m

Si noti che p e q non possono dividere entrambi m perché si avrebbe $m \geq n$ contro l'ipotesi sulla dimensione dei blocchi.

Generazione di un primo p grande

Ci sono tavole di numeri primi ma non così grandi...
Quanti numeri primi ci sono ? Ovvero, come sono distribuiti ? Qual è la probabilità di prendere un numero a caso e questo sia primo ?

$$Pr(n\ primo) \simeq \frac{1}{\log(n)}$$

Generazione di un primo p grande

n di 10 cifre: $Pr(n\ primo) = \frac{1}{23}$
n di 100 cifre: $Pr(n\ primo) = \frac{1}{230}$

Idea: Genero n a caso poi verifico se è primo. Se è primo ho finito. Altrimenti Inizio da capo.

Test di primalità

Fino a qualche tempo fa non si conosceva nessun algoritmo per testare se un numero è primo oppure no. Adesso esiste un algoritmo che svolge questo compito in tempo polinomiale.

Polinomiale ma poco efficiente. Come si procede in pratica ?

Teorema di Fermat

Se n è primo e $0 < a < n$ allora

$$a^{n-1} \mod n = 1$$

Pomerance 1981

Sia a numero casuale tra 1 e $n-1$
Sia n numero casuale di circa 100 cifre

$$Pr[(n\ non\ primo)\ e\ (a^{n-1} \mod n = 1)] \simeq \frac{1}{10^{13}}$$

Test di primalità probabilistico

1. Genero a tra 1 e n a caso
2. Calcolo $x = a^{n-1} \mod n$
3. Se $x = 1$ allora mi fermo e dichiaro che n è primo
4. Se $x \neq 1$ allora mi fermo e dichiaro che n è composto

Test di primalità probabilistico

Se n è primo: risposta esatta sempre.
Sia n numero casuale di circa 100 cifre
Se n è composto: risposta errata con probabilità $\simeq \frac{1}{10^{13}}$

La probabilità di errore $\frac{1}{10^{13}}$ è troppo alta ?

Come procediamo ?

Test di primalità probabilistico esteso

1. $k =$ numero intero positivo
2. Genero a tra 1 e n a caso
3. Calcolo $x = a^{n-1} \mod n$
4. Se $x \neq 1$ allora mi fermo e dichiaro che n è composto
5. Se $k > 0$ allora $k = k - 1$ e torno al punto 2
6. Mi fermo e dichiaro che n è primo

$$Pr(errore) = \left(\frac{1}{10^{13}}\right)^k$$

Come generare e e d

Il numero e deve essere primo con $\Phi(n)$.
Scelgo e a caso e verifico se è primo con $\Phi(n)$ con l'algoritmo di Euclide.
Se si, fine. Altrimenti ripeto il procedimento.

Il numero d deve essere l'inverso di e modulo $\Phi(n)$.
Con l'algoritmo di Euclide Esteso trovo l'inverso moltiplicativo di e modulo $\Phi(n)$.

Operazione di codifica e decodifica

Dobbiamo calcolare $a^b \mod c$ con a, b, c interi molto grandi.
Algoritmo stupido: Moltiplico a per a per a .... b volte e poi divido per c e prendo il resto.

Tempo di esecuzione dell'algoritmo stupido:
a e b di 150 cifre e usando tutti i calcolatori del mondo impieghiamo un tempo pari alla vita dell'universo !!!!

Operazione di codifica e decodifica

$$a \to a^2 \to a^4 \to a^8 \to a^{16} \cdots \to a^{32}$$

Cosa succede se devo calcolare a^{32+16} ?

Operazione di codifica e decodifica

Supponiamo di dover calcolare $123^{54} \mod 678$
Scriviamo l'esponente in binario e otteniamo $54 = 110110$
e poi calcoliamo in successione (applicando il modulo a ogni risultato intermedio)

123^{1}
123^{11}
123^{110}
123^{1101}
123^{11011}
123^{110110}

Operazione di codifica e decodifica

$$123^{1} = 123$$
$$123^{11} = 123^{2} \times 123$$
$$123^{110} = (123^{2} \times 123)^{2}$$
$$123^{1101} = ((123^{2} \times 123)^{2})^{2} \times 123$$
$$123^{11011} = \cdots$$
$$123^{110110} = \cdots$$

Operazione di codifica e decodifica

Le operazioni di modulo le distribuisco ad ogni passo (per ridurre le dimensioni dei risultati intermedi)

Il numero di operazioni è lineare nella lunghezza dell'esponente.

Algoritmo di Euclide per il Massimo Comun Divisore

$$MCD(a, b) = MCD(b, a \ mod \ b)$$

EUCLIDE-GCD(a, b)

```
1  if b == 0
2        return a
3  else return EUCLIDE-GCD(b, a mod b)
```

Algoritmo di Euclide esteso

```
EUCLIDE-ESTESO(a, b)
1   if b == 0
2         return (a, 1, 0)
3   else
4         (d', x', y') = EUCLIDE-ESTESO(b, a mod b)
5         return (d', y', x' − ⌊a/b⌋ y')
```

Algoritmo di Euclide esteso

supponiamo (chiamata ricorsiva) che
$d' = bx' + (a \mod n)y'$
e dimostriamo che
$d' = ay' + b(x' - \lfloor a/b \rfloor y')$

Algoritmo di Euclide esteso

$$
\begin{aligned}
bx' + (a \mod n)y' &= ay' + b(x' - \lfloor a/b \rfloor y') \\
&= bx' + ay' - b\lfloor a/b \rfloor y' \\
&= bx' + (a - b\lfloor a/b \rfloor)y' \\
&= bx' + (a \mod n)y'
\end{aligned}
$$

Algoritmo di Euclide esteso

$d = a\,x + b\,y$ (identità di Bézout)
se $d = 1$ allora $1 = a\,x + b\,y$ e quindi $a\,x = 1 - b\,y$ e
dunque $a\,x \mod b = 1$
x è l'inverso moltiplicativo di a modulo b

EUCLIDE-ESTESO(a, b)

```
1  (d, x, y) = EUCLIDE-ESTESO(a, b)
2  return x
```

Algoritmi di Euclide: costi

EUCLIDE-GCD(a, b) ha un costo $O(\log b)$

EUCLIDE-ESTESO(a, b) ha un costo $O(\log b)$

Cifrari ibridi

I cifrari asimmetrici sono "lenti" mentre i cifrari simmetrici obbligano a possedere chiavi segrete condivise. Per questo i due tipi di cifrari vengono in genere utilizzati in combinazione dando origine a cifrari ibridi in cui un cifrario a chiave pubblica è utilizzato per lo scambio della chiave segreta che viene impiegata nelle successive comunicazioni simmetriche.